Approximate Ad-hoc Query Engine for Simulation Data[i]

Ghaleb Abdulla, Chuck Baldwin, Terence Critchlow, Roy Kamimura , Ida Lozares, Ron Musick, NuAi Tang
CASC, Lawrence Livermore National Laboratory
Livermore, CA 94551
{abdulla1, baldwin5, critchlow, kamimura1, ilozares, tangn}@llnl.gov

Byung S. Lee, Robert Snapp
Department of Computer Science
University of Vermont
Burlington, VT 05405
(802)656-1919
{bslee,snapp}@cs.uvm.edu

ABSTRACT
In this paper, we describe AQSim, an ongoing effort to design and implement a system to manage terabytes of scientific simulation data. The project is aimed at reducing data storage requirements and access time while permitting ad-hoc queries using statistical and mathematical models of the data instead of the original data sets. In order to facilitate data exchange between models based on different representations, we are evaluating using the ASCI common data model which is comprised of several layers of increasing semantic complexity. To support queries over the spatial-temporal mesh structured data we are in the process of defining and implementing a grammar for MeshSQL

KEYWORDS
Mesh data, Scientific Data Management (SDM), Visualization, Data Integration, Query, Data Retrieval.

1. INTRODUCTION
Scientific data is commonly represented as a mesh. Mesh data is one of the most basic conceptual models for describing physical systems within computer models. A mesh breaks a surface or a volume down into an interconnected grid of smaller zones, each storing a set of computed variables. If the zones are small enough, the micro-scale properties and interactions can be modeled with sufficient accuracy to provide sufficient predictions of macro scale events. Storage and computation power requirements, however, increases with the number of zones. Current capabilities are at the scale of a few billion zones; a more typical range is between tens of thousands, to tens of millions of zones. For an elaborate description of the simulation mesh data please refer to [1].

Querying tera-scale data requires addressing several research challenges including the size of the data, multiple data formats, and supporting complex spatio-temporal queries. The data is generated using simulation software that takes weeks if not months for each run. Saving these data sets for query processing is not an option because of storage limitations. We are pursuing a multi-pronged approach to these issues. First, to minimize storage requirements we generate mathematical models of the data. Currently we generate a statistical summary and a wavelet model of the data for each mesh partition. Second, we use metadata associated with these models to facilitate the ad-hoc queries. This metadata help in matching the user query to the appropriate model to generate the most accurate answer within a user-specified error tolerance. We use the term "approximate" for the ad-hoc queries because of the described constraints. Since obtaining a highly accurate response can require a significant time, we provide the capability to trade accuracy for response time. Third, we are evaluating a mathematical model that will take into account the relationship between physical systems and mathematics. It considers the relationship between common mathematical entities in simulation and discrete representations of them employed in computer algorithms. This model can be exploited for data management and, in particular, we will use it for query optimization purposes [2].

The rest of the paper will describe the current system architecture and the research challenges.

2. SYSTEM ARCHITECTURE

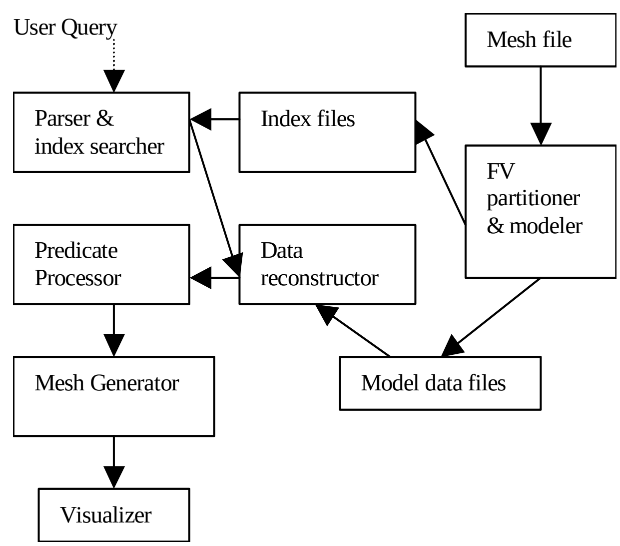

Figure 1 A simplified diagram of the current system architecture

Figure 1 shows a simplified diagram of the current architecture. On the right side of the diagram, we start with a mesh file, which is used to create a matrix of Feature Vectors (FVs), which is a mathematical vector containing all the spatio-temporal data of a given node in the mesh. The matrix is then partitioned into smaller sets based on a partition metric, generating an index tree of nodes. Each node may have one or more models. Each model in turn, will generate metadata, describing the model parameters, model error, model type along with other metadata attributes, all of which will be written out to the model data files. The final outcome of this initial phase is the generation of an index file describing the partitions and their associated model data files.

The generated files are smaller in size, however, they retain the information content of the original data at different resolutions. Those files will be used for the query processing while the original data is moved to tertiary storage.

A key research issue in this area is determining which partitioning and modeling techniques to use as well as interaction effects. Specifically, a variety of models will be examined - some more for their data compression capability others for their ability to address a wide variety of queries. Optimal model performance may be impacted by the partitioning scheme selected and this interaction needs to be examined. The current partitioning uses an octree-like structure using the spatial and time coordinates as inputs (see Figure 2). We are using the current partitioning method as at test-bed to evaluate the initial set of models, which include wavelets and b-splines:

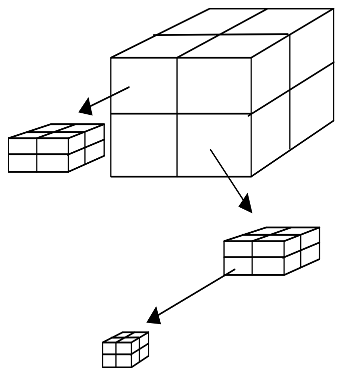

Figure 2 The partition algorithm.

The left side of Figure 1 shows the query engine, from which queries are entered in SQL syntax. The query statement is passed to a parser, and the resulting predicate is used by the index searcher to locate a set of candidate partitions. The partitions are then passed to the Data Reconstructor (DR), which uses model information to reconstruct data points in the required partition. The predicate processor evaluates the user query against these points and creates mesh data that can be viewed by a visualization application. The user query can include functions that use several variables to generate an implicit relationship. The DR uses the information from the user query, metadata, and the error tolerance specified by the user to pick the suitable model for data reconstruction.

3. RESEARCH CHALLENGES

3.1 THE QUERY ENGINE

Scientists often want to perform complex analysis of their data, not just perform simple selections over it, so the ability to include user-defined functions as part of the predicate is required. This increases the complexity of the query engine and requires supporting queries that do not include explicit relationships between variables. The solution is to capture and characterize as much as we can about the models, and heuristically define mappings from queries to the models that can provide for the best answer.

3.2 ERROR METRICS AND PARTITIONING ALGORITHMS

Since not all queries need to be highly accurate, we will investigate the relationship between the expected error, query speed, and the use of different levels of multi-resolution models. We will perform a series of experiments to determine the difference between the time for theoretical cost models and the observed time. We will use the results to improve error metrics (partition and model errors) at the nodes. We will fine-tune the models and the retrieval algorithms based on the obtained results.

3.3 DATA REDUCTION AND MODELING TECHNIQUES

Different data models are capable of answering different types of queries with different speeds. For example, users might be interested in following a certain region of the data over a certain period. Currently we are using statistical summary and wavelets to model the data. The statistical model allows us to quickly generate the data points, which can then be used to directly answer simple range queries. The Wavelet model allows us to easily identify areas of high variability, at varying resolutions, which are often of interest to the scientists.

We will be adding other modeling techniques in the future as needed.

4. CONCLUSIONS

We are currently developing the prototype as described in this paper. Initial results are promising and prove the concepts introduced here. Results show that we can create a system that will support approximate ad-hoc queries over large data sets. Because of this work we will be able to save the time and space needed to create and maintain large simulation data sets.

5. REFERENCES

[1] R. Musick, T. Critchlow, "Practical Lessons in Supporting Large-Scale Computational Science", SIGMOD Record Vol 28 Num 4, Dec 1999.

[2] J. Ambrosiano, D. M. Butler, C. Matarazzo, M. Miller "Development of a Common Data Model for Scientific Simulations," LANL Internal report, http://www.ca.sandia.gov/ASCI/cgi-bin/sdmframedisplay.cgi/ASCI/sdm/CDM.html.

[3] Data Engineering Bulletin: Special Issue on data reduction techniques, Vol. 20, No. 4, Dec 1997.

[i] This work was performed under the auspices of the U. S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract No. W-7405-ENG-48. UCRL-JC-138042